Bolt Beranek and Newman Inc.

Report No. 3809

Distributed Computation and TENEX-Related Activities

Final Report 1 November 1976 to 31 January 1978

May 1978

Prepared for:
Defense Advanced Research Projects Agency

BBN Report No. 3809

DISTRIBUTED COMPUTATION AND TENEX-RELATED ACTIVITIES

Final Report
1 November 1976 to 31 January 1978

Sponsored by:
Defense Advanced Research Projects Agency
ARPA Order No. 2935

Monitored by:
Office of Naval Research
Under Contract No. N00014-75-C-0773
Contract Period 1 November 1974 to 1 May 1978

Principal Investigators: Robert H. Thomas, Richard E. Schantz

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: BBN Report No. 3809

2. GOVT ACCESSION NO. (14) BBN-3809

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): (6) DISTRIBUTED COMPUTATION AND TENEX-RELATED ACTIVITIES.

5. TYPE OF REPORT & PERIOD COVERED: 11/1/76 - 1/31/78

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s): R. Schantz, R. Thomas

8. CONTRACT OR GRANT NUMBER(s): (15) N00014-75-C-0773, VARPA Order-2935

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
Bolt Beranek and Newman Inc.
50 Moulton Street
Cambridge, Massachusetts 02138

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS

12. REPORT DATE: (11) May [redacted]78

13. NUMBER OF PAGES: 30

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): (10) Richard E./Schantz, Robert H./Thomas

15. SECURITY CLASS. (of this report): Unclassified (12) 34 p.

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report):
Distribution of this document is unlimited. It may be released to the Clearinghouse, Department of Commerce for sale to the general public.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report): (9) Final rept. 1 Nov 76-31 Jan 78,

18. SUPPLEMENTARY NOTES
This research was supported by the Defense Advanced Research Projects Agency under ARPA Order No. 2935.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

distributed computation
National Software Works
distributed operating system
TENEX operating system

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
This report describes BBN efforts in the design of the National Software Works system and BBN efforts to integrate TENEX into the National Software Works system.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

060 100

TABLE OF CONTENTS

1. Introduction

This report is both the last quarterly progress report and the final report for Contract No. N00014-75-C-0773, Distributed Computation and TENEX-Related Activities. The contract covered BBN's National Software Works Project development efforts. The progress report portion is Sections 2 through 4. It covers the period from November 1977 to January 1978. The final report portion is Section 5. It summarizes project activity for the period from November 1976 to January 1978.

The major aspects of our NSW work for this quarter and for much of the contract period have focussed primarily on the tasks associated with developing an operational NSW system. By this we mean an NSW with sufficient functionality, adequate responsiveness and resiliency to transient failures of system components to be used on a daily basis for software production. The NSW has not yet completely achieved these goals, but significant advances toward them have been made during the contract period.

The following three sections discuss our efforts during the last quarter. Section 2 details improvements to the TENEX and TOPS-20 implementations of MSG, the interprocess communication facility for NSW. Our work on the Foreman for TENEX and TOPS-20, the TBH (tool bearing host) component which controls tool

execution in NSW is reported in Section 3. Improvements to other NSW related software are described in Section 4. The final section briefly highlights the NSW work performed during the last year, much of which has been described in detail in previous quarterly reports. (see BBN Reports No. 3736, 3751, 3752, and 3753).

In addition to our NSW implementation efforts, we participated in a number of meetings during the quarter. We also had an article published in the January 1978 issue of the IEEE Computer magazine. The article, entitled "Operating Systems for Computer Networks" discusses NSW as a network operating system and compares it with RSEXEC, another network operating system.

In November we attended a two day NSW project meeting held at Massachusetts Computer Associates in Wakefield. The topics discussed included further clarification of the long-term NSW reliability plan and the efforts relating to improving NSW performance. There was also extensive discussion regarding configuration management which is becoming increasingly important to the NSW project. Configuration management is concerned with the orderly evolution of a changing system. A configuration management plan includes clearly defined procedures for incorporating changes into the system, ranging from simple bug fixes to entire functional reorganizations. The discussion at the meeting focussed on establishing a set of different NSW

system configurations, each designated for somewhat different functions. Among the systems being considered are: a "debugging system" for use by all contractors to test and debug their components; a "test system" to be used for extensive regression testing of an integrated system about to be released to the user community; and a "user system" for well tested and documented system components in a generally stable user environment. The impact that these multiple NSW system configurations might have on the NSW contractors, as well as proposals for procedures guiding the migration of changes through the systems, were discussed in detail.

Following the contractors meeting there was an NSW project review for the sponsoring agencies. For this review, we summarized the major BBN NSW developments since the last review in December 1976. Our presentation covered the major achievements and short-term goals for our efforts on the TENEX and TOPS-20 implementations of the MSG and Foreman components. The meeting also established the project's resolve to address problems relating to system performance and to add a network mail capability during the forthcoming year. A number of other organizational and facilities management issues were raised and discussed.

We also participated in other meetings related to NSW this past quarter. At the request of the ARPA office we have been

meeting with a group from Harvard University regarding their ideas for automating certain aspects of software system development and maintenance. The intent here is to use MSG as a test case for evaluating these ideas. The discussions are chiefly serving to familiarize the Harvard group with MSG and to increase our understanding of their work. Finally, we consulted with the computer center staff from the Rome Air Development Center regarding the facility management of their newly acquired TOPS-20 configuration. That computer installation is currently projected as a prominent part of a future operational NSW.

2. MSG Developments

Major improvements were made this quarter to the TENEX and TOPS-20 implementations of MSG in the area of host configuration control. At the MSG level, "configuration control" mechanisms enable an NSW system operator to specify the components which are to coordinate their activity to provide NSW service.

One part of MSG configuration control consists of restricting the hosts with which processes may communicate to those hosts specified when the NSW configuration is defined by the system operator. Whenever the host control facility is enabled, message addresses are checked to ensure that the host field designates a legitimate recipient for the intended configuration. The configuration is defined as those hosts listed in the host specification file, augmented with those hosts specified by the operator during the MSG startup dialog. Messages addressed to hosts which are not part of the specified configuration are not delivered and the sending processes are signalled appropriately. This host configuration control facility is controlled by a runtime switch internal to MSG. When enabled, it provides a more secure environment for running multi-host systems such as NSW and, in addition, it can be used as an aid in debugging part of the process MSG interface.

Another aspect of MSG configuration control deals with handling local host resource allocation associated with MSG generic addressing. We have expanded the flexibility with which the generic classes can be supported, and have simplified the operator interface for specifying the desired MSG behavior for each generic type. These changes are part of an on-going overall effort that we have undertaken to make it easier for a configuration to be tailored to the needs of a particular environment.

Some of these improvements became high priority tasks when MSG was converted to run under TOPS-20. Due to the absence of certain operating system features, it became infeasible to use the existing allocation strategies for some NSW components. Specifically, the strategy used by MSG to allocate TENEX Foreman processes (the CLASSJOB Locate-Spec, see below) depends upon the TENEX directory fork group feature to ensure that different Foreman processes in the same job do not interfere with each other. Because directory fork groups are not supported on TOPS-20, MSG must use a different allocation mechanism to prevent interference between Foreman processes. This mechanism was provided by adding a new option for allocating processses to handle generically addressed messages (the NEWJOB Create-Spec, see below) to MSG's repetoire. The particular strategy used by MSG for a given process class is controlled by the system operator at configuration definition time.

Another improvement in this area makes it possible for an NSW operator to exert control over the number of processes MSG allocates in each generic process class. Previously, the process allocation/deallocation done by MSG was quite simple. Roughly speaking, new processes would be created to receive generically addressed messages whenever there was no process in the class with a pending receive operation, and processes would be deallocated (killed) only if the process requested to be terminated and the operator had specified the KILLPROC Terminate-Spec (see below). In practice KILLPROC is seldom used in NSW and as a result, the number of processes in an NSW MSG configuration can only grow. The current version of MSG corrects this problem by allowing the operator to specify for each generic class the maximum number of processes that MSG should allocate at any time. In addition, he may instruct MSG to pre-allocate a specified number of processes.

Finally, in the area of host configuration control, it is now possible to define generic process classes that are supported on several different hosts. When a process sends a generically addressed message without host specification to such a class, MSG selects a host to send the message to. Internal to MSG there is a routine that selects the host given the process class. At present, the selection criteria is simple; the first host on a list of hosts is selected. However, the selection strategy can

easily be changed by modifying this internal MSG routine. This routine will be the basis for future MSG development in the area of global resource allocation strategies for handling generically addressed messages.

As a result of these local host configuration control improvements, we can now support TENEX and TOPS-20 generic classes with sufficient generality to handle all of the anticipated needs of the NSW project.

The following describes the current specification format and options associated with local host support for generic process classes.

The generic name file is a text file. It consists of a list of name definitions, each of which is a line of the form:

Name Code Create-Spec Terminate-Spec Saved-File-Name

or:

Name Code Create-Spec Host-Spec.

The first form is used to define locally supported processes: "Name" is the generic class being defined; "Code" is the MSG-to-MSG internal code for the generic class (an integer <128.); "Create-Spec" specifies how generically addressed messages for the class are to be handled by MSG when there are no

outstanding ReceiveGenericMessage primitives by existing processes in the class; "Terminate-Spec" defines how the StopMe primitive is to be handled when a process in the class executes it; "Saved-File-Name" is the name of the saved file for the process core image.

The second form is used to define remotely implemented processes. Here "Create-Spec" must be the string "REMOTEJOB"; "Host-Spec" is an expression of the form:

Host [, Host]*

where Host is either the ARPANET name (a text string) or the ARPANET address (an octal integer) for the host which supports the process class.

Create-Spec is an expression of the form:

CSpec [, CModifier]* .

At present the following CSpecs are defined:

CLASSJOB - All processes in the class are to execute in a TENEX job dedicated to that class.

MISCJOB - All processes in the class are to execute in a TENEX job dedicated to "miscellaneous" processes.

NEWJOB - Each process in the class is to execute in a TENEX job by itself.

REMOTEJOB - Processes in this class execute on the specified host(s).

SINGLEPROC - At most, only a single process of this type may exist at any time.

The CModifier is used to control the action taken by the central MSG with respect to a given generic process class at initialization time. The following CModifiers are defined:

INITGM - When MSG is started, a null generic message is to be sent (from MSG) to a process in the class. A null generic message can be recognized as a message whose length is zero and whose sender is the null process; the null process name is a process name for which bytes 1-7 are zero.

INITJOB (or INITJOB=N) - To be used with the NEWJOB CSpec. When MSG is started, create the specified number (N) of jobs to support the process class. If N is not specified, N=1 is assumed. If INITJOB is not specified, N=0 is assumed if the NOJOB CModifier is used and N=1 is assumed if the JOB CModifier is used.

INITPROC (or INITPROC=N) - To be used with the CLASSJOB CSpec. When MSG is started, create the specified number (N) of processes for the process class. If N is not specified, N=1 is assumed. If INITPROC is not specified, N=0 is assumed.

JOB - MSG is to create and start a process-controlling MSG job for the class at initialization time. This is the default CModifier which is always assumed whether specified or not unless overridden by the NOJOB CModifier.

NOJOB - MSG should not create a process-controlling job for the class at initialization time.

The NOJOB modifier would be an appropriate CModifier (to the NEWJOB CSpec) for NSW Front End (Front End) processes which are started up by the NSW dispatcher. In addition, it should be useful in a debugging environment where it is desirable to have a process class defined at initialization by the central MSG but run under

programmer and debugger control via a manually started process-controlling MSG.

Terminate-Spec is an expression of the form:

TSpec [, TModifier]* .

At present the following TSpecs are defined:

KILLPROC - When the process executes StopMe, kill it (via the TENEX KFORK JSYS).

RESTARTPROC - When the process executes StopMe, assign a new MSG process name to the TENEX fork(s) which implements it and restart the fork at its start address.

STOPMSG - Same as KILLPROC with the exception that if no more processes exist in the job, the process-controlling MSG will terminate.

The TModifier is used to further control the action taken by MSG when a process in the class is terminated. The following TModifiers are currently defined:

MAXJOB (or MAXJOB=N) - To be used with the NEWJOB CSpec. MSG is to support at most N jobs for this process class. If more than N exist when a process is terminated, the process-controlling MSG for that job will stop regardless of the TSpec; i.e., it will act as if the TSpec were STOPMSG. If N is not specified, N=1 is assumed.

MAXPROC (or MAXPROC=N) - To be used with the CLASSJOB CSpec. MSG is to support at most N processes of this class. If more than N exist when a process is terminated, the fork(s) which implement that process will be killed regardless of the TSpec; i.e., MSG will act as if the TSpec were KILLPROC. If N is not specified, N=1 is assumed.

In an MSG related area, we have programmed, tested and released an updated version of the module which provides a programmable interface to the ARPANET user TELNET function. This module is used by the NSW Front End, in conjunction with MSG direct connection primitive operations, to handle Front End communication with most NSW tools. The major improvement incorporated in this release is the addition of code to generate the TELNET "synch sequence." The TELNET synch sequence is used to support a terminal "attention" or interrupt mechanism. It is generated by sending a host/host protocol INS interrupt for the TELNET connection and inserting a TELNET Data Mark command to mark the place in the data stream where the interrupt occurred. Since the INS signal is not subject to flow control, it is used to alert the receiving process to immediately scan the data stream (up to and including the Data Mark) for "important" TELNET commands. Such commands may include Interrupt Process (IP), a function generally used to regain control of a "runaway" process, and Abort Output (AO), a function used to discard accumulated output. Some ARPANET hosts, including the IBM 360/91 at UCLA which is now becoming an interactive tool bearing host, require the synch signal to ensure proper processing of some TELNET commands. To accommodate these hosts, the user TELNET module now automatically includes the synch sequence for those TELNET commands which the Front End has previously instructed it to do so. This change should make it possible to more easily deal with the UCLA machine as a tool bearing host.

3. Foreman Developments

During the past quarter we prepared and released a new TENEX Foreman component. The new Foreman incorporates a number of improvements in performance and instrumentation, in addition to fixing all known problems. According to our preliminary measurements, the CPU time required for the Foreman role in the BEGINTOOL scenario (i.e., starting a tool session) has been reduced by about 30% from our last release.

The new Foreman provides two types of logging functions. The first type supports on-line observation of NSW operation (Foreman event recording and message interpretation) and creates a record of errors encountered in running the Foreman. The error file is created only if an error occurs and all Foremen from a given host NSW incarnation place their error reports in this single file. This mode of error recording makes it very convenient for an operator to visually determine whether or not a period of NSW activity induced error conditions, and if so, all Foreman error reports can be processed from a single file.

The second type of logging function supports the measurement of tool bearing host performance, Foreman resource utilization, and tool usage and encapsulation requirements. It also documents the particular Front End/Foreman configuration under which the statistics are gathered to permit comparative testing of various

local-Front End and remote-Front End NSW configurations. The statistics gathering Foreman instrumentation uses the general purpose measurement and analysis software whose development we reported in our previous QPR (BBN Report No. 3753).

There were also a number of minor functional refinements in the new Foreman. These include a feature whereby the Foreman discards a saved tool session which is older than a specified age (currently five days). A saved tool session is one which was not terminated under normal conditions. A user can "rerun" a saved tool session to retrieve any files which were in the workspace when the tool ceased to be operational. The current implementation approach taken in the TENEX and TOPS-20 tool bearing host software is to maintain saved tool sessions in the same "space" used to support active tool sessions. Because the space available for tool sessions is limited, as the number of saved sessions grows the number of active tools that can be supported decreases. Saved sessions are discarded after an appropriate period as a temporary measure to free workspaces for use with new tool sessions. A more permanent solution to the workspace reclamation problem has been designed (see Appendix A of BBN Report No. 3752) but has not as yet been implemented.

The Foreman can now recover from and repair a defective list of tool sessions which are being held for rerunning. Prior to this release, a Foreman which crashed while updating this list

could under certain circumstances leave the data base in an unuseable state.

Finally, the TENEX tool encapsulation code has been upgraded to heuristically handle the TENEX feature for requesting programmable file name recognition (Control F in GTJFN) by the operating system. The problem arises when integrating DEC supplied tools, which support names with a maximum of six characters, with TENEX tools (no effective restriction on name length) and attempting to support them in NSW which does not support file name recognition at all.

A major part of last quarter's Foreman development effort was spent converting the TENEX Foreman to run under TOPS-20. As with the MSG conversion, we are taking the approach of developing a single load module which at runtime can detect the type of system (TENEX or TOPS-20) on which it is running under and execute operating system dependent code where necessary. The major advantage of this approach is that it greatly simplifies maintenance of the Foreman software; there is no need to manage seperate source and object files for the two systems, and core images can be freely moved between systems. The disadvantage, which appears to be minor, is a slight increase in the number of instructions executed in handling those functions which differ on the two systems.

The Foreman has been completely converted to run on both TENEX and TOPS-20 release 1. However, the testing and checkout of the Foreman running on TOPS-20 cannot be completed as yet because of the unavailability of a TOPS-20 File Package. When the File Package becomes operational, we can then complete the testing of the integrated TOPS-20 software and release a system which can include TOPS-20 as a tool bearing host.

Part of the effort of converting the Foreman for use on TOPS-20 has been in adapting the system call transformations associated with tool encapsulation to the new environment. To validate these modifications, we have installed and tested (to the extent possible without a File Package) TOPS-20 versions of most currently available TENEX NSW tools in our test configuration. Some tools have not as yet been converted to TOPS-20 for direct use by TOPS-20 users and are consequently not available through NSW.

To aid in the functional testing of NSW, and to help drive the NSW sessions we use for performance measurement, we have developed the NSW scripts to be used with a TENEX version of the RITA command processing system. RITA (the Rand Intelligent Terminal Agent) is a set of computer programs designed to allow the efficient development of "user agents" comprised of pattern replacement rules expressed in a simple English-like language. The NSW oriented RITA agents execute scripts calling for logging

into NSW, running tools, accessing files and logging off. These are all accomplished without human intervention. Now that we have established the rules for carrying out many of the NSW functions by the user agent, we are turning our attention to the problem of developing an agent which exercises as much of the Foreman functionality as is possible from the user interface. The intent here is to establish and automate a series of tests which the Foreman must pass before it will be released as a new component to the larger NSW community.

The test encapsulator is an interactive TENEX program we previously developed to help with tool encapsulation and installing additional NSW tools. It traps a set of JSYSs executed by the program running under it similar to those trapped by the NSW Foreman for an encapsulated tool. The test encapsulator allows the user to obtain control whenever a trap occurs to examine the state of the primitive operation and the tool. We use it extensively to determine a tool's NSW related operating system interface without the overhead of actually running an NSW configuration. From these sessions we can often anticipate potential problems with encapsulating the tool in question.

This quarter we modified the test encapsulator so that it runs on the TOPS-20 system. Like our other NSW related components, there is a single executable file for the test

encapsulator, which determines at runtime the type of system and adapts accordingly. We have also started an effort to have the program parse the parameters associated with certain key JSYS calls, and report these variables to the user in an English-like fashion. For example, when the GTJFN JSYS, which is used to get a handle on a file given its name, is trapped, the test encapsulator now automatically prints out the file name, key word indicators relating to the nature of the file being sought (e.g., new file only, old file only), and other pertinent parameters specifically relating to the function of the call. This information supplements the normal printout of the numerical contents of the low order accumulator registers which normally hold system call parameters. This symbolic printout makes it much easier to extract the pertinent system call data and also provides a means for automatically annotating a hard copy record of the tool's operating characteristics. The raw numeric parameter data is still available for those parameters which are not yet parsed by the test encapsulator. Ultimately, we hope to have all of the important NSW related JSYSs interpreted symbolically by the test encapsulator program. Other improvements to the test encapsulator include complete recognition of all TENEX/TOPS-20 JSYS by mneumonics in addition to JSYS number, and the ability to resume execution of the trapped fork at various offsets from the point of trapping. This latter facility, in conjunction with the ability to modify the

fork's registers and address space from within the test encapsulator after a call has been trapped, allows the user to simulate alternative system call interpretations. It also allows us to exercise the tool's code for handling non-standard return patterns, again without the overhead of actually running an NSW configuration.

4. Other NSW Related Software Developments

In addition to our development and maintenance responsibilities for the MSG and Foreman NSW components, BBN has also developed and maintains other NSW related software modules. These include the NSW Front End Dispatcher, which automatically connects an NSW user to a Front End job in response to a network request; the MKCOM program, which is used to create and manage the files associated with NSW accounts and directories need to operate as a tool bearing host; and the general purpose measurement analysis packages used for component instrumentation. There have been developments this quarter with each of these programs.

The Dispatcher module was modified to accommodate the needs of the SRI NSW development team. SRI has been working on the problem of installing NLS, a sophisticated text manipulation system, into NSW as a tool whose functionality is implemented cooperatively by the Front End and the tool process. Our Foreman support for this effort, mostly in the area of enhanced tool communication facilities, was reported last quarter. To properly demonstrate the NLS NSW capability, SRI requested a Dispatcher with properties somewhat different from the standard NSW module. These changes included different contact addresses (sockets) to distinguish from other Front Ends which are not able to support NLS, a different type of user connection to the Front End (i.e.,

different Telnet mode), and different program configuration characteristics to better suit an operator-less environment. The changes to satisfy their immediate requirements were incorporated into a special Dispatcher to be used for their demonstration purposes. These changes should also prove valuable for future Front End development flexibility and are now being incorporated on a permanent basis into the Dispatcher component for general use by NSW.

Both the MKCOM program and the general instrumentation package have been converted to be operational on TOPS-20 configurations using system release 1. This is part of the general NSW project effort to make TOPS-20 first operational as a tool bearing host and then to operationally support the Works Manager and Front End functions also. Accordingly, we have immediate plans to convert the Dispatcher to the TOPS-20 environment.

5. Summary of Contract Activity

There has been substantial progress on the NSW project in general, and on BBN related NSW efforts in particular during the past year. As part of this final report, we will briefly highlight our major accomplishments during the contract period, and indicate where the NSW stands today. Most of the items summarized here have been reported on in more depth in the various quarterly progress reports. We refer the reader to BBN reports 3736, 3751, 3752, 3753 and the preceding sections of this report for additional details.

The major achievement of the current NSW effort has been the transformation of the NSW from a demonstration system to a prototype operational capability. At the start of the period, the system was brought up by hand only for demonstration purposes and then only to perform carefully selected operations which were known not to break the system. At present there is an NSW which is generally available on a daily service basis with few restrictions on using the defined functionality. To be sure, the NSW has not reached the point where it can be considered to be a finished product. Despite significant activity during the year directed toward improving performance and reliability, the system does require additional improvements in these areas to be considered totally operational. These efforts, as well as efforts to increase the available functionality through

additional hosts, tools and commands, are part of the continuing NSW development plan.

For a number of years now, BBN has played a dual role in the NSW project. We have been part of the design team specifying the software architecture, and we have also been responsible for developing and maintaining the software to integrate TENEX and TOPS-20 into NSW as tool bearing hosts. The NSW functional design concepts are recorded in a series of component specification documents. We have written two of these documents and this year have updated them to reflect recent design activity. The updated documents are available as "MSG: The Interprocess Communication Facility for the NSW" (BBN Report No. 3473) and "The Foreman: Providing the Program Execution Environment for the NSW" (BBN Report No 3442). Additionally, we have produced and continue to update an "MSG User's Manual" (BBN Report No. 3540) which is used not only by other NSW contractors but also in other projects using MSG for communication support.

Throughout the contract period there were numerous new releases of the system components which we implement. The increased availability of NSW has led to the discovery, isolation and correction of many of the bugs which are part of every large, complex system. This series of new component releases have been part of the systematic "hardening" of the NSW; today new "bug reports" are relatively infrequent. When bugs do occur they are

most often associated with newer functional additions. This phase of removing the operational problems through actual use is one which most development projects must endure, and certainly has contributed to the confidence with which one can invoke the various NSW functions. The new releases also bring the implementations up to date with regard to the component specification in most areas.

In addition to removing bugs, other aspects of NSW system reliability have and continue to be emphasized. These range from improving the ability of components to individually cope with transient resource failures to overall coordinated efforts at providing reliable system behavior using distributed components. An example of the former is MSG's ability to recover from the unexpected termination of ARPANET connections that support MSG-to-MSG communication. An important quality of our recent component releases is their ability to continue operating despite certain failures in their operating environment. Error detection and reporting facilities have been added and upgraded to better inform other components of the detected failures, and also to provide facilities for alerting human users and operators to the problem. The intercomponent protocols have been augmented to incorporate more comprehensive error reporting, including human oriented text strings indicating the failure. Additionally, and perhaps most significantly, there has been an extensive effort

placed in defining what reliable NSW operation will actually mean in terms of supportable functions, and component scenarios were developed to support this type of reliable behavior. These are the so-called interim and full reliability scenarios. In general, they are designed with the goal of protecting completed user activity and ensuring the integrity of system data bases in the presence of component outages. After helping to develop the plan, we implemented for the TENEX tool bearing host the interim reliability scenarios which make it possible to retrieve user files that were trapped in prematurely terminated tool sessions. As a by product of that development, we are now faced with the problem of managing the files associated with these "saved" tool sessions until they are retrieved and moved into the NSW file system. We have designed, but not yet implemented, a flexible facility to handle this problem (BBN Report No. 3752, Appendix A).

Performance is another key area which has received much attention during the past year. It is clear from using NSW that there are considerable problems with system performance. Even under the best of circumstances NSW does not perform adequately to be considered a legitimate, operational entity. What is unclear is the causes for the performance problems.

In an effort to find system bottlenecks, we have begun an intensive effort to instrument the system in order to measure

the performance of its various parts. We have instrumented MSG to record both its internal performance in handling the various MSG activities and the various performance characteristics of the NSW components which run under its control. The occurence of the various MSG interactions initiated by the NSW components is logged. With a knowledge of the NSW operations being performed and the appropriate NSW inter-component protocols, the log can be analyzed to extract performance measures for the components executing their parts of variosus NSW scenarios. Experiments have been devised and run to measure MSG's operating performance characteristics under a variety of internal data base structures and under a variety of conditions.

We have also extensively instrumented the TENEX Foreman component both to measure its internal performance and to record the frequency and real time cost of the various NSW operations as viewed from the Foreman. This aspect of data collection is usually active for the operational user NSW system, so we have an accurate picture of how frequently various NSW requests are made and the response time provided under a realistic environment. As part of our overall performance related effort we have distributed two full reports on NSW performance measurements (BBN Report No. 3751, Appendix A and BBN Report No. 3753, Section 4) and have released new system components with improved performance characteristics based on obtainable measures. We are also

working closely with a group from the University of Texas at Austin in developing an overall approach toward improved NSW performance.

Partly as a result of our efforts, the functionality supported by NSW has greatly increased during the past year. We have made a number of additional TENEX utility programs available as NSW tools. These include: FTP, a program for invoking the ARPANET File transfer mechanism to import and export NSW files; SPELL, a spelling corrector program; BCPL, a high level system programming language processor; BDDT, a high level debugging package for BCPL programs; LINKER, a program for creating TENEX executable "run" modules; and JIGSAW, an interactive JOVIAL programming language preprocessor. Interesting things to note with regard to tools is that NSW now supports a complete BCPL programming environment for PDP-10s (in addition to the previously supported programming environment for UYK-20s), and a tool such as JIGSAW can now be offered in both its batch oriented form through the MULTICS tool bearing host and an interactive form by TENEX. We have also developed a new and important NSW facility through a tool we call DESCRIBE. The DESCRIBE tool is an on-line help facility which provides NSW users with pertinent information on using the available NSW tool set. Although currently rather primitive, DESCRIBE fills an important void in the NSW functionality, and we expect that it will lead to the more extensive help facility needed for a successful NSW.

Another important activity started during this contract period but not yet totally complete is the integration of the TOPS-20 system into NSW as a tool bearing host. TOPS-20 is a descendent of the TENEX operation system. However, it is not identical to TENEX. As a result, each TENEX NSW component needs to be converted to run under TOPS-20. To accomplish this conversion, we first specified additions to the TOPS-20 operating system required to make it functionally equivalent to current versions of TENEX in areas which are of particular concern for distributed systems such as NSW. As part of another contract we implemented these system modifications, and they are now part of the standard version if the TOPS-20 system released and maintained by DEC.

Both MSG and the Foreman have been converted to run under TOPS-20 release 1 software. To make TOPS-20 operational as an NSW tool bearing host, Massachusetts Computer Associates must convert the TENEX File Package component to run under TOPS-20. Current NSW project plans include moving the Works Manager and Front End components from TENEX to TOPS-20 because TOPS-20 is a more cost effective machine than the older KA-10 machines which support TENEX. However, additional conversion efforts for the Works Manager, Front End, and Dispatcher are required to accomplish the full transfer of TENEX NSW software to TOPS-20.

Over the course of the year we have developed software to support testing, error detection and easy operation of the system and its individual components. The following highlights some of these operational aids. MSG has been equipped with a process monitoring and debugging capability supporting a variety of informational and action oriented commands. Included are commands for setting a minimum timeout interval for MSG operations and for forcing the immediate timeout of an event whose timeout interval has not yet elapsed. Both of these are extremely valuable in testing and diagnostic procedures. The new Foreman instrumentation package includes message decoding and recording logic and event recording primitives geared to on-line debugging of the Foreman component. Both are text oriented and can be switched on or off as the need arises. Both MSG and the Foreman support error detection mechanisms which can be used by operations personnel to help pinpoint error conditions. We now provide very flexible configuration setup and management with MSG through the use of conveniently updated text files. Other operator and maintenance functions, such as tool workspace reclamation, have been automated to make it easier for an NSW system operator to manage various system resources. We have developed programs for the off-line processing of the event log files created by MSG and the Foreman to make the data convenient to use without undue on-line processing overhead. We have also developed a test encapsulator program that enables us to evaluate

tool-to-host-system interactions related to NSW tool encapsulation without requiring a full NSW configuration. All of these and other related developments are oriented toward the efficient integration of new and upgraded system components and features, and in smoothly and conveniently operating the resulting facility.

Finally, an additional part of our NSW activities dealt with familiarizing other organizations with NSW activities and determining areas of mutual interest. Discussions have been held on using the NSW or parts of it (e.g., MSG as a communication facility) within other ARPA sponsored projects such as the ACCAT facility. We have also lectured and written about the NSW project in an effort to expose the concepts to a wider audience. In the long run, these activities may also help to accelerate the progress of NSW into a commonly accepted utility.

As this summary of our efforts indicates, there has been substantial progress on the NSW development project over the last year. However, we believe that additional work is required to achieve the goal of a fully operational facility.